Zabbix Manual

Welcome to the user manual for Zabbix 2.0 software. These pages are created to help our users successfully manage their monitoring tasks with Zabbix, from the simple to the more complex.

2.0/manual.txt · Last modified: 2012/05/23 11:27 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Zabbix definitions

Overview

In this section you can learn the meaning of some terms commonly used in Zabbix.

Definitions

host

- a networked device that you want to monitor, with IP/DNS.

host group

- a logical grouping of hosts; it may contain hosts and templates. Hosts and templates within a host group are not in any way linked to each other. Host groups are used when assigning access rights to hosts for different user groups.

item

- a particular piece of data that you want to receive off of a host, a metric of data.

trigger

- a logical expression that defines a problem threshold and is used to "evaluate" data received in items

When received data are above the threshold, triggers go from 'Ok' into a 'Problem' state. When received data are below the threshold, triggers stay in/return to an 'Ok' state.

event

- a single occurrence of something that deserves attention such as a trigger changing state or a discovery/agent auto-registration taking place

action

- a predefined means of reacting to an event.

An action consists of operations (e.g. sending a notification) and conditions (*when* the operation is carried out)

escalation

- a custom scenario for executing operations within an action; a sequence of sending notifications/executing remote commands

media

- a means of delivering notifications; delivery channel

notification

- a message about some event sent to a user via the chosen media channel

remote command

- a pre-defined command that is automatically executed on a monitored host upon some condition

template

- a set of entities (items, triggers, graphs, screens, applications, low-level discovery rules) ready to be applied to one or several hosts

The job of templates is to speed up the deployment of monitoring tasks on a host; also to make it easier to apply mass changes to monitoring tasks. Templates are linked directly to individual hosts.

application

- a grouping of items in a logical group

web scenario

- one or several HTTP requests to check the availability of a web site

frontend

- the web interface provided with Zabbix

Zabbix API

- Zabbix API allows you to use the JSON RPC protocol to create, update and fetch Zabbix objects (like hosts, items, graphs and others) or perform any other custom tasks

Zabbix server

- a central process of Zabbix software that performs monitoring, interacts with Zabbix proxies and agents, calculates triggers, sends notifications; a central repository of data

Zabbix agent

- a process deployed on monitoring targets to actively monitor local resources and applications

Zabbix proxy

- a process that may collect data on behalf of Zabbix server, taking some processing load off of the server

node

- a full Zabbix server configured as an element within a hierarchy of distributed monitoring; it is responsible for monitoring its own location

2.0/manual/concepts/definitions.txt · Last modified: 2012/04/02 13:24 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Server

Overview

Zabbix server is the central process of Zabbix software.

The server performs the polling and trapping of data, it calculates triggers, sends notifications to users. It is the central component to which Zabbix agents and proxies report data on availability and integrity of systems. The server can itself remotely check networked services (such as web servers and mail servers) using simple service checks.

The server is the central repository in which all configuration, statistical and operational data is stored, and it is the entity in Zabbix that will actively alert administrators when problems arise in any of the monitored systems.

The functioning of a basic Zabbix server is broken into three distinct components; they are: Zabbix server, web frontend and database storage.

All of the configuration information for Zabbix is stored in the database, which both the server and the web frontend interact with. For example, when you create a new item using the web frontend (or API) it is added to the items table in the database. Then, about once a minute Zabbix server will query the items table for a list of the items which are active that is then stored in a cache within the Zabbix server. This is why it can take up to two minutes for any changes made in Zabbix frontend to show up in the latest data section.

Server process

Zabbix server runs as a daemon process. The server can be started by executing:

```
shell> cd sbin
shell> ./zabbix_server
```

You can use the following command line parameters with Zabbix server :

```
-c --config <file>              absolute path to the configuration file (default is /etc/zabbix/zabbix_server.conf)
-n --new-nodeid <nodeid>        convert database data to new nodeid
-R --runtime-control <option>   perform administrative functions
-h --help                       give this help
-V --version                    display version number
```

Runtime control is not supported on OpenBSD and NetBSD.

Examples of command line parameters:

```
shell> zabbix_server -c /usr/local/etc/zabbix_server.conf
shell> zabbix_server --help
shell> zabbix_server -V
```

Runtime control

Runtime control options:

Option	Description
config_cache_reload	Reload configuration cache. Ignored if cache is being currently loaded.

Example of using runtime control to reload the server configuration cache:

```
shell> zabbix_server -c /usr/local/etc/zabbix_server.conf -R config_cache_reload
```

Process user

Zabbix server is designed to run as a non-root user. It will run as whatever non-root user it is started as. So you can run server as any non-root user without any issues.

If you will try to run it as 'root', it will switch to a hardcoded 'zabbix' user, which must be present on your system.

These settings currently cannot be user configured, neither during compilation nor in the configuration file.

If Zabbix server and agent are run on the same machine it is recommended to use a different user for running the server than for running the agent. Otherwise, if both are run as the same user, the agent can access the server configuration file and any Admin level user in Zabbix can quite easily retrieve, for example, the database password.

Configuration file

See the configuration file options for details on configuring zabbix_server.

Start-up scripts

The scripts are used to automatically start/stop Zabbix processes during system's start-up/shutdown. The scripts are located under directory misc/init.d.

Supported platforms

Due to the security requirements and mission-critical nature of server operation, UNIX is the only operating system that can consistently deliver the necessary performance, fault tolerance and resilience. Zabbix operates on market leading versions.

Zabbix server is tested on the following platforms:

- Linux
- Solaris
- AIX
- HP-UX
- Mac OS X
- FreeBSD
- OpenBSD
- NetBSD
- SCO Open Server
- Tru64/OSF1

Zabbix may work on other Unix-like operating systems as well.

(linking to other sections, like zabbix maintenance etc)

2.0/manual/concepts/server.txt · Last modified: 2012/07/19 10:52 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Agent

Overview

Zabbix agent is deployed on a monitoring target to actively monitor local resources and applications (hard drives, memory, processor statistics etc).

The agent gathers operational information locally and reports data to Zabbix server for further processing. In case of failures (such as a hard disk running full or a crashed service process), Zabbix server can actively alert the administrators of the particular machine that reported the failure.

Zabbix agents are extremely efficient because of use of native system calls for gathering statistical information.

Passive and active checks

Zabbix agents can perform passive and active checks.

In a passive check the agent responds to a data request. Zabbix server (or proxy) asks for data, for example, CPU load, and Zabbix agent sends back the result.

Active checks require more complex processing. The agent must first retrieve a list of items from Zabbix server for independent processing. Then it will periodically send new values to the server.

Whether to perform passive or active checks is configured by selecting the respective monitoring item type. Zabbix agent processes items of type 'Zabbix agent' or 'Zabbix agent (active)'.

Supported platforms

Zabbix agent is supported for:

- Linux
- IBM AIX
- FreeBSD
- NetBSD
- OpenBSD
- HP-UX
- Mac OS X
- Solaris
- Windows: 2000, Server 2003, XP, Vista, Server 2008, 7

Installation

See installation instructions for Zabbix agent.

Agent process for UNIX (standalone daemon)

Zabbix agent is run on the host being monitored. Zabbix agent runs as a daemon process.

To start the agent, execute:

```
shell> cd sbin
shell> ./zabbix_agentd
```

The following command line parameters can be used with Zabbix agent:

```
  -c --config <file>   specify configuration file, default is /etc/zabbix/zabbix_agentd.conf
  -h --help            give this help
  -V --version         display version number
  -p --print           print known items and exit
  -t --test <item key> test specified item and exit
```

For example, to get this help, run:

```
shell> zabbix_agentd -h
```

Other examples of command line parameters:

```
shell> zabbix_agentd -c /usr/local/etc/zabbix_agentd.conf
shell> zabbix_agentd --help
shell> zabbix_agentd --print
shell> zabbix_agentd -t "system.cpu.load[all,avg1]"
```

Process user

Zabbix agent is designed to run as a non-root user. It will run as whatever non-root user it is started as. So you can run agent as any non-root user without any issues.

If you will try to run it as 'root', it will switch to a hardcoded 'zabbix' user, which must be present on your system. You can only run agent as 'root' if you modify the 'AllowRoot' parameter in the agent configuration file accordingly.

Configuration file

See the configuration file options for details on configuring zabbix_agentd.

Running agent for Windows

See Zabbix Windows agent installation for the installation information.

To start the agent service you can use Control Panel or run:

```
zabbix_agentd.exe --start
```

Command line syntax:

```
zabbix_agentd.exe [-Vhp] [-idsx] [-c <file>] [-t <metric>]
```

The following command line parameters can be used with Zabbix Windows agent.

Options:

```
  -c --config <file>   Specify alternate configuration file (default is c:\zabbix_agentd.conf).
```

```
  -h --help            Display help information.
  -V --version         Display version number.
  -p --print           Print known items and exit.
  -t --test <item key> Test single item and exit.
```

Functions:

```
  -i --install   Install Zabbix agent as a service.
  -d --uninstall Uninstall Zabbix agent service.
  -s --start     Start Zabbix agent service.
  -x --stop      Stop Zabbix agent service.
```

Configuration file

See the configuration file options for details on configuring Zabbix Windows agent.

2.0/manual/concepts/agent.txt · Last modified: 2013/06/06 08:45 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 Proxy

Overview

Zabbix proxy is a process that may collect monitoring data from one or more monitored devices and send the information to the Zabbix server, essentially working on behalf of the server. All collected data is buffered locally and then transferred to the Zabbix server the proxy belongs to.

Deploying a proxy is optional, but may be very beneficial to distribute the load of a single Zabbix server. If only proxies collect data, processing on the server becomes less CPU and disk I/O hungry.

A Zabbix proxy is the ideal solution for centralized monitoring of remote locations, branches and networks with no local administrators.

Zabbix proxy requires a separate database.

The databases that are supported with Zabbix proxy are SQLite, MySQL and PostgreSQL. Using Oracle or IBM DB2 is at your own risk and may contain limitations as is the case with low-level discovery rules.

See also: Using proxies in a distributed environment

Proxy process

Zabbix proxy runs as a daemon process. The proxy can be started by executing:

```
shell> cd sbin
shell> ./zabbix_proxy
```

You can use the following command line parameters with Zabbix proxy:

```
-c --config <file>              absolute path to the configuration file
-R --runtime-control <option>   perform administrative functions
-h --help                       give this help
-V --version                    display version number
```

Examples of command line parameters:

```
shell> zabbix_proxy -c /usr/local/etc/zabbix_proxy.conf
shell> zabbix_proxy --help
shell> zabbix_proxy -V
```

Runtime control

Runtime control options:

Option	Description
config_cache_reload	Reload configuration cache. Ignored if cache is being currently loaded. Active Zabbix proxy will connect to the Zabbix server and request configuration data.

Example of using runtime control to reload the proxy configuration cache:

```
shell> zabbix_proxy -c /usr/local/etc/zabbix_proxy.conf -R config_cache_reload
```

Runtime control is not supported on OpenBSD and NetBSD.

Process user

Zabbix proxy is designed to run as a non-root user. It will run as whatever non-root user it is started as. So you can run proxy as any non-root user without any issues.

If you will try to run it as 'root', it will switch to a hardcoded 'zabbix' user, which must be present on your system.

These settings currently cannot be user configured, neither during compilation nor in the configuration file.

Configuration file

See the configuration file options for details on configuring zabbix_proxy.

2.0/manual/concepts/proxy.txt · Last modified: 2012/07/19 11:00 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

5 Java gateway

Overview

Zabbix 2.0 adds native support for monitoring JMX applications by introducing a new Zabbix daemon called "Zabbix Java gateway". Zabbix Java gateway is a daemon written in Java. When Zabbix server wants to know the value of a particular JMX counter on a host, it asks Zabbix Java gateway, which uses the JMX management API [http://java.sun.com/javase/technologies/core/mntr-mgmt/javamanagement/] to query the application of interest remotely. The application does not need any additional software installed, it just has to be started with `-Dcom.sun.management.jmxremote` option on the command line.

Java gateway accepts incoming connection from Zabbix server or proxy and can only be used as a "passive proxy". As opposed to Zabbix proxy, it may also be used from Zabbix proxy (Zabbix proxies cannot be chained). Access to each Java gateway is configured directly in Zabbix server or proxy configuration file, thus only one Java gateway may be configured per Zabbix server or Zabbix proxy. If a host will have items of type **JMX agent** and items of other type, only the **JMX agent** items will be passed to Java gateway for retrieval.

When an item has to be updated over Java gateway, Zabbix server or proxy will connect to the Java gateway and request the value, which Java gateway in turn retrieves and passes back to the server or proxy. As such, Java gateway does not cache any values.

Zabbix server or proxy has a specific type of processes that connect to Java gateway, controlled by the option **StartJavaPollers**. Internally, Java gateway starts multiple threads, controlled by the **START_POLLERS** option. On the server side, if a connection takes more than **Timeout** seconds, it will be terminated, but Java gateway might still be busy retrieving value from the JMX counter.

Zabbix server or proxy will try to pool requests to a single JMX target together as much as possible (affected by item intervals) and send them to the Java Gateway in a single connection for better performance.

It is suggested to have **StartJavaPollers** less than or equal to **START_POLLERS**, otherwise there might be situations when no threads are available in the Java gateway to service incoming requests.

Sections below describe how to get and run Zabbix Java gateway, how to configure Zabbix server (or Zabbix proxy) to use Zabbix Java gateway for JMX monitoring, and how to configure Zabbix items in Zabbix GUI that correspond to particular JMX counters.

5.1 Getting Java gateway

There are two ways to get Java gateway. One is to download Java gateway package from Zabbix website and the other is to compile Java gateway from source.

5.1.1 Downloading from Zabbix website

This option is not actually available yet, but in future you will be able to download Java gateway archive from Zabbix website.

5.1.2 Compiling from source

In order to compile Java gateway, you first run `./configure` script with `--enable-java` option. It is advisable that you specify `--prefix` option to request installation path other than the default /usr/local, because installing Java gateway will create a whole directory tree, not just a single executable.

```
$ ./configure --enable-java --prefix=$PREFIX
```

To compile and package Java gateway into a JAR file, run `make`. Note that for this step you will need `javac` and `jar` executables in your path.

```
$ make
```

Now you have zabbix-java-gateway-$VERSION.jar file in src/zabbix_java/bin. If you are comfortable with running Java gateway from src/zabbix_java in the distribution directory, then you can proceed to instructions for configuring and running Java gateway. Otherwise, make sure you have enough privileges and run `make install`.

```
$ make install
```

5.2 Overview of files in Java gateway distribution

Regardless of how you obtained Java gateway, you should have ended up with a collection of shell scripts, JAR and configuration files under $PREFIX/sbin/zabbix_java. The role of these files is summarized below.

```
bin/zabbix-java-gateway-$VERSION.jar
```

Java gateway JAR file itself.

```
lib/logback-core-0.9.27.jar
lib/logback-classic-0.9.27.jar
lib/slf4j-api-1.6.1.jar
lib/org-json-2010-12-28.jar
```

Dependencies of Java gateway: Logback [http://logback.qos.ch/], SLF4J [http://www.slf4j.org/], and JSON.org [http://www.json.org/] library.

```
lib/logback.xml
lib/logback-console.xml
```

Configuration files for Logback.

```
shutdown.sh
startup.sh
```

Convenience scripts for starting and stopping Java gateway.

```
settings.sh
```

Configuration file that is sourced by startup and shutdown scripts above.

5.3 Configuring and running Java gateway

By default, Java gateway listens on port 10052. If you plan on running Java gateway on a different port, you can specify that in settings.sh script. See the description of Java gateway configuration file for how to specify this and other options.

Port 10052 is not IANA registered [http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.txt].

Once you are comfortable with the settings, you can start Java gateway by running the startup script:

```
$ ./startup.sh
```

Likewise, once you no longer need Java gateway, run the shutdown script to stop it:

```
$ ./shutdown.sh
```

Note that unlike server or proxy, Java gateway is lightweight and does not need a database.

5.4 Configuring server for use with Java gateway

Now that Java gateway is running, you have to tell Zabbix server where to find Zabbix Java gateway. This is done by specifying JavaGateway and JavaGatewayPort parameters in server configuration file. If the host on which JMX application is running is monitored by Zabbix proxy, then you specify the connection parameters in proxy configuration file instead.

```
JavaGateway=192.168.3.14
JavaGatewayPort=10052
```

By default, server does not start any processes related to JMX monitoring. If you wish to use it, however, you have to specify the number of pre-forked instances of Java pollers. You do this in the same way you specify regular pollers and trappers.

```
StartJavaPollers=5
```

Do not forget to restart server or proxy, once you are done with configuring them.

5.5 Debugging Java gateway

In case there are any problems with Java gateway or an error message that you see about an item in the frontend is not descriptive enough, you might wish to take a look at Java gateway log file.

By default, Java gateway logs its activities into /tmp/zabbix_java.log file with log level “info”. Sometimes that information is not enough and there is a need for information at log level “debug”. In order to increase logging level, modify file lib/logback.xml and change the level attribute of <root> tag to “debug”:

```
<root level="debug">
  <appender-ref ref="FILE" />
</root>
```

Note that unlike Zabbix server or Zabbix proxy, there is no need to restart Zabbix Java gateway after changing logback.xml file - changes in logback.xml will be picked up automatically. When you are done with debugging, you can return the logging level to “info”.

If you wish to log to a different file or a completely different medium like database, adjust logback.xml file to meet your needs. See Logback Manual [http://logback.qos.ch/manual/] for more details.

Sometimes for debugging purposes it is useful to start Java gateway as a console application rather than a daemon. To do that, comment out PID_FILE variable in settings.sh. If PID_FILE is omitted, startup.sh script

starts Java gateway as a console application and makes Logback use lib/logback-console.xml file instead, which not only logs to console, but has logging level “debug” enabled as well.

Finally, note that since Java gateway uses SLF4J for logging, you can replace Logback with the framework of your choice by placing an appropriate JAR file in lib directory. See SLF4J Manual [http://www.slf4j.org/manual.html] for more details.

2.0/manual/concepts/java.txt · Last modified: 2011/11/09 19:15 by richlv

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

6 Sender

Overview

Zabbix sender is a command line utility that may be used to send performance data to Zabbix server for processing.

The utility is usually used in long running user scripts for periodical sending of availability and performance data.

Running Zabbix sender

An example of running Zabbix UNIX sender:

```
shell> cd bin
shell> ./zabbix_sender -z zabbix -s "Linux DB3" -k db.connections -o 43
```

where:

- z - Zabbix server host (IP address can be used as well)
- s - monitored host name (as registered in Zabbix front-end)
- k - item key
- o - value to send

See Zabbix sender manpage for more information.

Zabbix sender accepts strings in UTF-8 encoding (for both UNIX-like systems and Windows).

Zabbix sender on Windows can be run similarly:

```
zabbix_sender.exe [options]
```

Since Zabbix 1.8.4, zabbix_sender realtime sending scenarios have been improved to gather multiple values passed to it in close succession and send them to the server in a single connection. A value that is not further apart from the previous value than 0.2 seconds can be put in the same stack, but maximum pooling time still is 1 second.

If sending many values from an input file, Zabbix sender will batch them at 250 values in one go (all values will be processed), for example:

```
# zabbix_sender -z 127.0.0.1 -i /tmp/traptest.txt
Info from server: "Processed 250 Failed 0 Total 250 Seconds spent 0.002668"
Info from server: "Processed 50 Failed 0 Total 50 Seconds spent 0.000540"
sent: 300; skipped: 0; total: 300
```

All entries from an input file are sent in a sequential order top-down.

If the target item has triggers referencing it, all timestamps in an input file must be in an increasing order, otherwise event calculation will not be correct.

Zabbix sender will terminate if invalid (not following *parameter=value* notation) parameter entry is present in

specified configuration file.

2.0/manual/concepts/sender.txt · Last modified: 2013/11/19 09:23 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

7 Get

Overview

Zabbix get is a process which can be used to communicate with Zabbix agent and retrieve required information from the agent.

The utility is usually used for the troubleshooting of Zabbix agents.

Running Zabbix get

An example of running Zabbix get under UNIX to get the processor load value from the agent:

```
shell> cd bin
shell> ./zabbix_get -s 127.0.0.1 -p 10050 -k "system.cpu.load[all,avg1]"
```

Zabbix get accepts the following command line parameters:

```
  -s --host <host name or IP>      Specify host name or IP address of a host.
  -p --port <port number>          Specify port number of agent running on the host. Default is 10050.
  -I --source-address <IP address> Specify source IP address.
  -k --key <item key>              Specify key of item to retrieve value of.
  -h --help                        Give this help.
  -V --version                     Display version number.
```

Zabbix get on Windows can be run similarly:

```
zabbix_get.exe [options]
```

2.0/manual/concepts/get.txt · Last modified: 2012/06/21 16:40 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]